Simulador de Aprendizado Federado e Análise de Convergência do Treinamento

Gabryel Medeiros de Oliveira, Lisandro Lovisolo,
Guilherme Lucio Abelha Mota e Helio do Nascimento Cunha Neto

***Resumo*— No paradigma de aprendizado federado, diversas fontes de dados são empregadas para aprender sobre um mesmo problema ou modelo. As instâncias são usadas para aprender/atualizar localmente o modelo. Cada atualização do modelo é enviada a um servidor central que atualiza o modelo global. O modelo global é reencaminhado para os dispositivos. Esse processo é iterado até atingir-se o critério de convergência. Este trabalho, apresenta um simulador de aprendizado federado. Tal ferramenta permite estudar e analisar impactos de diferentes parâmetros do treinamento, número de máquinas e outros aspectos na convergência do aprendizado. A ferramenta pretende ser uma plataforma de testes para a avaliação de diferentes formas de quantizar, codificar e aleatorizar as atualizações para avaliar seus impactos no aprendizado e na segurança do processo de treinamento, de forma a obter uma codificação com algum nível de privacidade.**

***Palavras-Chave*— Aprendizado Federado, Aprendizado de Máquina, Convergência**

***Abstract*— In the federated learning paradigm, multiple data sources collaborate to solve a single task or train a shared model. Each device uses its local data to train or update the model independently, and the resulting updates are transmitted to a central server, which aggregates them to refine the global model. The updated global model is then redistributed to the devices. This cycle is repeated until a predefined convergence criterion is satisfied. This work introduces a federated-learning simulator that allows researchers to investigate how various training parameters, the number of participating machines, and other factors influence convergence. The simulator is designed as a testbed for assessing different quantization, encoding, and randomization strategies applied to model updates, thereby enabling the study of their impact on both learning performance and training-phase security, with the aim of achieving an update encoding that affords a meaningful degree of privacy.**

***Keywords*— Federated Learning, Machine Learning, Convergence**

I. Introdução

O Aprendizado Federado (*Federated Learning* — FL) surgiu como uma alternativa promissora para treinar modelos de aprendizado de máquina de forma decentralizada [1]. Em FL, cada dispositivo treina localmente o seu modelo e compartilha apenas parâmetros atualizados com um servidor, denominado servidor de agregação, mantendo os dados brutos onde foram gerados [2]. Esse paradigma descentralizado preserva a privacidade dos dados, reduzindo riscos de descumprimento de leis de proteção de dados, como a Lei Geral de Proteção de Dados (LGPD) [3].

Gabryel Medeiros de Oliveira, Guilherme Lúcio Abelha Mota, Helio do Nascimento Cunha Neto, Instituto de Matemática e Estatística, UERJ, e-mail: oliveira.gabryel@graduacao.uerj.br, guimota@ime.uerj.br, helio.cunha@ime.uerj.br; Lisandro Lovisolo, Laboratório de Processamento de Sinais, Aplicações Inteligentes e Comunicações (PROSAICO), UERJ, e-mail: lovisolo@eng.uerj.br.

Em FL, a principal fonte de tráfego ocorre durante a agregação — o ciclo em que o servidor distribui o modelo global, os clientes o ajustam localmente e devolvem as atualizações para agregação. Esse intercâmbio de vetores de parâmetros pode atingir centenas de megabytes por rodada [1]. O problema torna-se crítico em ambientes heterogêneos, nos quais os dispositivos variam em largura de banda e latência: *smartphones* ligados a redes 2G/3G, sensores de Internet das Coisas (*Internet of Things* — IoT) com Wi-Fi e estações cabeadas de alta velocidade colaborando no mesmo treinamento. Nessas condições, enlaces lentos atrasam as rodadas de agregação e diminuem a taxa de convergência, exigindo técnicas que reduzam o volume de dados sem comprometer a qualidade do modelo.

Este artigo propõe um simulador de aprendizado federado que instrumenta a camada de comunicação, assim, permite inserir estratégias de codificação e quantização e medir o tráfego gerado por cada cliente e os impactos no aprendizado. A ferramenta foi concebida para acelerar a prototipação de técnicas de redução de custo de comunicação. Como estudo de caso, emprega-se o simulador para investigar como diferentes tamanhos de mini-lote influenciam o custo total de comunicação e o número de rodadas necessárias para convergência, demonstrando que escolhas de hiperparâmetros podem impactar significativamente a eficiência de rede em ambientes heterogêneos.

A análise experimental revelou que o ajuste do tamanho do *mini-batch*, como a escolha de um *mini-batch* de tamanho 12 em vez de 3, proporcionou uma redução de aproximadamente 75% no tráfego total transmitido, mantendo a acurácia final em cerca de 90,5%. Esta acurácia situa-se dentro de aproximadamente 2,8% (ou uma queda de menos de 3 pontos percentuais) em relação à acurácia máxima de 93,1% obtida com o *mini-batch* de tamanho 3. Esses resultados reforçam a importância de considerar conjuntamente parâmetros de treinamento e estratégias de comunicação no projeto de sistemas federados.

O restante deste artigo está organizado da seguinte forma: A Seção II introduz uma breve descrição dos princípios do FL. A Seção III descreve a arquitetura do simulador. A Seção IV discute os experimentos e resultados. Por fim, a Seção V conclui o trabalho e apresenta direções para pesquisas futuras a partir do simulador apresentado.

II. Aprendizado Federado

O Aprendizado Federado (FL) realiza múltiplas iterações de treinamento local nos dispositivos clientes antes da agregação das atualizações no servidor central [2]. Essa

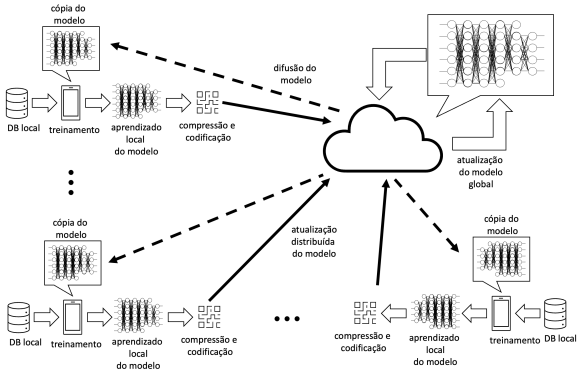

Fig. 1
Modelo federado para ANNs: cada dispositivo calcula gradientes locais, que podem ser comprimidos e enviados ao servidor para atualizar o modelo global, depois redistribuído.

abordagem implica no custo de comunicação associado à transmissão das atualizações do modelo (sejam parâmetros ou gradientes). Esse é um gargalo crítico, especialmente em cenários com modelos de grande porte ou um número expressivo de dispositivos participantes.

A Figura 1 resume essa abordagem. Basicamente, os dispositivos federados produzem a partir de bases de dados distintas, atualizações locais do modelo. Por sua vez, o dispositivo central recebe as atualizações e as combina. O ciclo finaliza quando a atualização global resultante para o modelo é comunicada aos dispositivos federados. Tal processo deve ser iterado até atingir-se a convergência.

Em um paradigma como o acima, dependendo da aplicação e do modelo há ganhos de segurança pois os dados em si não são enviados, somente as atualizações do modelo são remetidas, dificultando violações de privacidade [4]. O envio das atualizações pode conferir o que se chama de privacidade diferencial (*differential privacy*) [5]. Porém, dependendo da granularidade do lote, do modelo e da codificação das atualizações (como elas são representadas e transmitidas), ainda assim pode haver alguma exposição com vazamento de informação [3].

Complementarmente, como o conjunto de dados em geral apresenta dimensões (instâncias$\times$características$\times$dispositivos) maiores que a do modelo, há um ganho de escala. A transmissão do modelo global atualizado exige banda de transmissão. Porém, em muitos casos, os requisitos de banda são menores que no aprendizado de máquina centralizado, coadunando com a assimetria geralmente observada em redes sem-fio nas taxas de *downlink* (da central para um dispositivo) e de *uplink* (o caminho inverso). Além disso, o uso de modos de transmissão *broadcast* (difusão) pode reduzir ainda mais essa demanda. De qualquer forma, tais ganhos são irrelevantes se o desempenho do modelo $\mathcal{W}^{(\mathrm{fed})}$ federado não for comparável ao do modelo centralizado.

Diversas estratégias têm tentado mitigar o custo de comunicação do FL. Dentre elas, destacam-se a redução da frequência de comunicação através de mais processamento local [2], a otimização do processo de agregação (e.g., [6]) e a seleção inteligente de clientes [7]. Dentre as múltiplas configurações que alteram o compromisso entre comunicação e computação localizada, os impactos do tamanho do *mini-batch* e do número de épocas locais no treinamento local, testados a partir do simulador apresentado, são fatores que podem afetar o compromisso entre convergência do modelo e requisitos de comunicação.

Técnicas de compressão de gradientes ou atualizações do modelo visam diminuir a quantidade de dados transmitidos em cada rodada de comunicação. No entanto, ao introduzirem aproximações ou perdas de informação, essas abordagens estabelecem uma relação de compromisso (*trade-off*) não-linear entre a redução no volume de dados e o potencial impacto na velocidade de convergência ou na acurácia final do modelo. Essas abordagens são construídas usando **quantização**, que reduz a precisão numérica dos gradientes, e **esparsificação**, que seleciona uma fração dos valores dos gradientes, para comunicação e cômputo de atualizações. [8] analisa diversas estratégias, incluindo esparsificação e quantização estruturada. O QSGD (*Quantized Stochastic Gradient Descent*) de [9] propõe uma quantização estocástica dos gradientes, e em [10], tem-se a quantização de gradientes usando três níveis, chamada de TernGrad. Outra abordagem de quantização extrema é o SignSGD, que utiliza apenas o sinal dos gradientes [11], reduzindo drasticamente a comunicação.

Visando otimizar a relação de compromisso entre compressão e desempenho, pesquisas mais recentes têm explorado técnicas de compressão mais sofisticadas. Algumas propostas focam em explorar a redundância temporal ou o histórico das atualizações, como a compressão de gradientes baseada em Wyner-Ziv [12]. Outras investigam a compressão baseada na dimensionalidade intrínseca dos gradientes [13]. A quantização vetorial, utilizando *codebooks* para representar os gradientes, também tem sido objeto de estudo [14], [15]. Abordagens híbridas, que combinam diferentes técnicas de compressão para otimizar o trade-off entre taxa de compressão e desempenho do modelo, continuam a ser desenvolvidas [16].

A variedade e complexidade das estratégias de otimização e compressão no FL requer ferramentas flexíveis para experimentação e avaliação de desempenho. Simuladores, como o desenvolvido e apresentado neste artigo, desempenham um papel fundamental na prototipagem rápida, análise de impacto de hiperparâmetros e na comparação de diferentes abordagens em cenários controlados, facilitando o avanço da área. Neste contexto, o simulador detalhado neste trabalho foi desenvolvido com o intuito de oferecer uma plataforma aberta (tanto no código como nos dados), ágil e controlável para a investigação do impacto de diversos parâmetros de treinamento e, crucialmente, para a prototipagem e avaliação de diferentes estratégias de codificação e quantização das atualizações de modelo em Aprendizado Federado, facilitando a análise de suas consequências na convergência e no custo de comunicação.

III. Simulador Proposto

Neste trabalho, com o propósito de examinar o comportamento do aprendizado federado sob diferentes estratégias de comunicação, foi implementado[1], um simulador (emulador) de aprendizado federado. O simulador foi desenvolvido em Python com base na biblioteca TensorFlow[2]. Esse simulador nos permite emular um sistema com um servidor central e múltiplos clientes, implementando diferentes estratégias de envio dos gradientes para investigar seu impacto no processo de aprendizado e avaliação de diferentes parâmetros de treinamento.

A biblioteca está estruturada em três componentes principais: o `CentralServer`, que coordena o aprendizado; os `Clients`, que representam os participantes federados com dados locais; e a `ExperimentalUnit`, que executa os experimentos federados. A Figura 2 apresenta o diagrama de classes do simulador, com os principais atributos e métodos de cada componente.

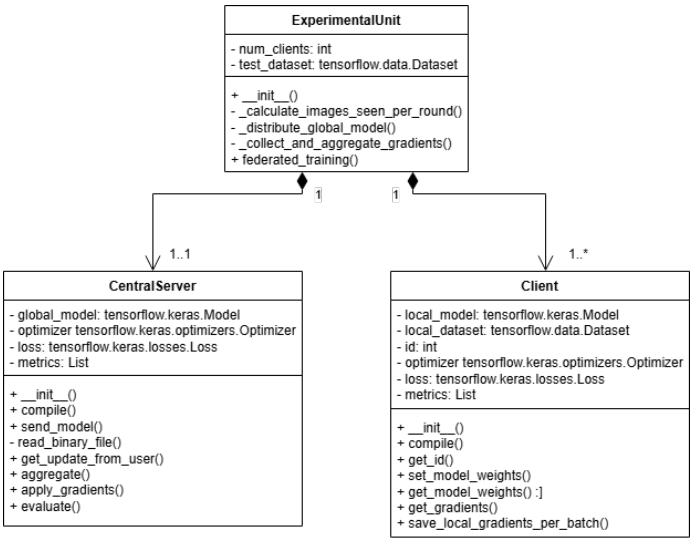

Fig. 2
Diagrama de classes do simulador de aprendizado federado.

A. Servidor Central (*CentralServer*)

O `CentralServer` coordena o treinamento, mantendo o modelo global e agregando as atualizações enviadas pelos clientes. O ciclo de comunicação é conduzido por meio da distribuição do modelo global (`send_model()`) e da coleta das atualizações de cada dispositivo (`get_update_from_user()`). A função crucial do servidor é `aggregate(updates)`, que combina as atualizações dos clientes (atualmente através da média simples). O modelo global é atualizado com os gradientes agregados usando `apply_gradients(aggregated_gradients)`, e seu desempenho é avaliado periodicamente em um conjunto de dados de teste (`evaluate(test_dataset)`).

B. Clientes Federados (*Client*)

Cada `Client` simula um participante com seus próprios dados de treinamento. Os clientes recebem o modelo global do servidor e realizam treinamento local. Implementamos duas estratégias para calcular e enviar atualizações: o envio de gradientes por lote dos seus dados locais, e o envio da média dos gradientes calculados sobre todo conjunto de dados local (isto é, lotes tão grandes quanto os dados). Os clientes possuem métodos para gerenciar seu modelo local e computar os gradientes para ambas as estratégias.

C. Unidade Experimental (*ExperimentalUnit*)

A `ExperimentalUnit` orquestra o experimento federado. Ela inicializa o servidor e os clientes, coordena as rodadas de treinamento e comunicação, e coleta as métricas de desempenho. A unidade experimental implementa o laço de treinamento federado, garantindo a correta interação entre o servidor e os clientes para a execução das diferentes estratégias de agregação e a avaliação do modelo global ao longo do tempo.

[1]Disponível em `https://github.com/gabryelmedeiros/FederatedLearning`.

[2]Disponível em `https://www.tensorflow.org/`. Acessado em 07/05/2025

IV. Experimentos e Análise

A. Aplicação: Classificação de Dígitos com MNIST

Para avaliar a eficácia das estratégias de comunicação implementadas, conduziram-se experimentos utilizando o conjunto de dados MNIST para reconhecimento de dígitos manuscritos. A arquitetura da rede neural convolucional desenvolvida combina princípios consolidados na literatura de visão computacional. O processamento inicia-se com uma camada convolucional (32 filtros 3×3, ReLU) que implementa o paradigma de extração hierárquica de características espaciais proposto por [17]. A subsequente operação de *max pooling* com janela 3×3 e *stride* 2, seguindo 18, introduz invariância a translações locais enquanto reduz dimensionalidade.

A transição para características semânticas de alto nível é realizada através de uma camada densa com 128 unidades (ReLU), estratégia alinhada com recomendações de [19] para combinação não-linear de atributos. A camada de saída emprega 10 neurônios com ativação *softmax*, padrão estabelecido para classificação multiclasse. A seleção de hiperparâmetros como tamanho de *kernel* e número de filtros otimiza relação de compromisso entre capacidade discriminativa e custo computacional, particularmente relevante para dados de baixa resolução como o MNIST.

O ambiente de simulação foi configurado com 10 clientes e foi executado em uma única máquina. O conjunto de 2.000 imagens do MNIST foi dividido entre os clientes sem controle explícito sobre a distribuição de dígitos. Essa divisão aleatória resulta em distribuições não-IID, onde os dados de cada cliente apresentam variações significativas na distribuição de dígitos [1]. Cada dispositivo passa a possuir um subconjunto de dados com vieses específicos de rótulos ou características, replicando assim uma propriedade fundamental dos ambientes reais do aprendizado federado.

Usamos 2.000 imagens para permitir comparações entre as estratégias, mantendo o custo computacional e comunicacional dentro de limites viáveis para o cenário proposto.

Como limitante de desempenho treinamos o modelo de forma centralizada, o que permite avaliar possíveis reduções no desempenho do treinamento centralizado, realizado com todos os dados disponíveis em um único local.

Foi aplicada validação cruzada com 5 folds, e os valores apresentados correspondem à média dos resultados, permitindo uma estimativa mais robusta do desempenho.

B. Eficiência da Transmissão vs. Desempenho de Aprendizado

A Figura 3 apresenta a acurácia do modelo global em função da quantidade acumulada de dados transmitidos (em megabytes) durante o treinamento federado para diferentes tamanhos de mini-batch. O critério de parada adotado foi o número total de amostras processadas, independentemente do número de rodadas de comunicação, garantindo comparabilidade entre os experimentos.

Observou-se que tamanhos de mini-batch maiores tendem a alcançar acurácias superiores com menor volume de dados transmitidos. Isso se deve ao fato de que, com mini-batches maiores, os clientes realizam mais iterações locais antes de cada transmissão, reduzindo a frequência de comunicação sem comprometer significativamente a qualidade da atualização global. Por outro lado, mini-batches pequenos, como 3, resultam em um elevado número de transmissões com pouca informação agregada por rodada, o que leva a uma convergência mais lenta e maior custo de comunicação.

A linha de base em vermelho representa o desempenho do treinamento centralizado, realizado com 2000 imagens ao longo de 10 épocas de treinamento. Essa abordagem atingiu acurácia próxima de 97%.

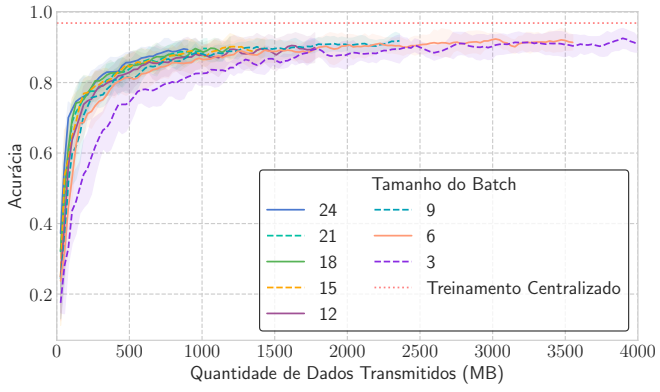

Fig. 3
Acurácia global em função do volume acumulado de dados transmitidos para diferentes tamanhos de batch.

C. Eficiência Computacional vs. Qualidade do Modelo

A Figura 4 mostra a evolução da acurácia do modelo global em função do número total de amostras processadas no treinamento. Essa métrica reflete diretamente o custo computacional total de cada abordagem, independentemente da frequência de comunicação entre os dispositivos.

A estratégia de transmissão da média dos gradientes, na qual os clientes não realizam atualizações locais e apenas transmitem o gradiente médio calculado através de todo o dataset local, apresenta desempenho inferior, revelando a importância das atualizações locais para a efetividade do aprendizado. Quanto à estratégia que emprega atualizações locais, a Figura 4 indica que, para o mesmo número total de amostras processadas, configurações com *mini-batches* menores tenderam a alcançar os melhores resultados de acurácia.

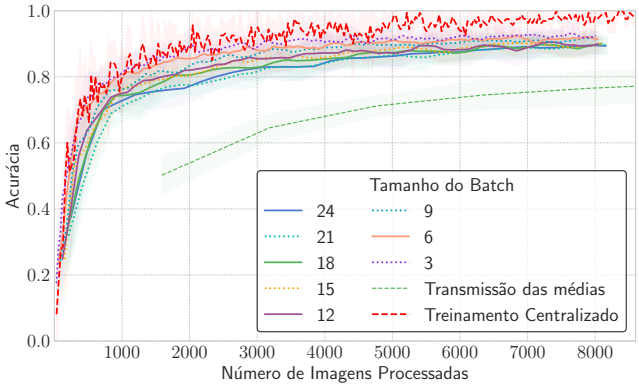

Fig. 4
Acurácia global em função do número total de amostras processadas durante o treinamento. Resultados obtidos via validação cruzada com 5 folds.

D. Discussão

A escolha do tamanho do *mini-batch* impacta diretamente a eficiência de comunicação no treinamento federado, conforme revelam os resultados (Figura 3). Observou-se que *mini-batches* de maior dimensão, utilizados durante o treinamento local em cada cliente, tendem a produzir gradientes locais menos ruidosos. Isso pode contribuir para uma convergência mais estável do modelo global, exigindo um menor número de rodadas de comunicação para se atingir uma determinada acurácia e, consequentemente, alcançaram acurácias elevadas com um menor volume total de dados transmitidos. Esta dinâmica está alinhada com as preocupações sobre a eficiência da comunicação em aprendizado descentralizado, onde o volume de dados trocados é um gargalo crítico, conforme extensivamente discutido por [2]. Em contraste, *mini-batches* menores, conforme observado experimentalmente, exigiram um volume de comunicação acumulado superior para atingir um nível de acurácia comparável.

A análise da Figura 4, que relaciona a acurácia com o número de imagens processadas, aponta para uma direção distinta da observada sob a ótica da eficiência de comunicação. Os resultados indicam que configurações com *mini-batches* menores - por exemplo, o tamanho 3 que alcançou

aproximadamente 93,1% de acurácia, em contraste com cerca de 89,5% para o tamanho 24 - atingiram os melhores níveis de acurácia final em comparação com *mini-batches* maiores nos testes federados. Este fenômeno pode ser atribuído à maior granularidade das atualizações. Com *mini-batches* menores, o modelo global recebe contribuições mais frequentes e baseadas em porções menores dos dados locais. Essa abordagem pode ser particularmente benéfica em ambientes com dados heterogêneos (não-IID), um desafio inerente ao aprendizado federado cujo impacto tem sido investigado por autores como [20]. A granularidade permite que o modelo global se ajuste de forma mais ágil e contínua às diversas distribuições de dados dos clientes, mitigando o risco de modelos locais divergirem excessivamente devido às particularidades de seus conjuntos de dados.

A abordagem de transmissão da média dos gradientes – na qual cada cliente calcula um único gradiente agregado sobre a totalidade de seus dados locais por rodada, – demonstrou desempenho inferior, conforme visualizado na Figura 4. Embora esta estratégia envolva o processamento de todo o conjunto de dados local de cada cliente a cada rodada para gerar sua contribuição, sua eficácia em termos de acurácia alcançada por volume de amostras processadas é notavelmente menor em comparação com as abordagens que empregam otimização local iterativa. Este resultado reforça a importância crucial das atualizações locais e do processamento iterativo com *mini-batches* para a eficácia do aprendizado federado, um princípio fundamental estabelecido nos trabalhos seminais de [2].

V. Conclusão e Trabalhos Futuros

Neste trabalho, apresentamos uma abordagem localizada para emular o aprendizado federado. Os testes realizados com uma aplicação em reconhecimento de dígitos indicam que o emulador funciona corretamente e permite avaliar os impactos da escolha de hiperparâmetros no processo de aprendizado federado. O simulador apresentado permite analisar as mensagens trocadas entre diferentes máquinas federadas para poder avaliar diferentes compromissos de desempenho de forma aberta. A principal motivação é testar estratégias de quantização e codificação das atualizações e modelos, de forma simples, para analisar estratégias e métodos para a redução da taxa e seus impactos no processo de aprendizado federado bem como de segurança dos usuários e máquinas envolvidos no processo. Em trabalhos futuros, diferentes estratégias de codificação e quantização dos pesos serão avaliadas em termos de taxa e impacto no aprendizado; dentre essas destacamos, a princípio, técnicas simples como o descarte de pesos em função de seus valores, o descarte aleatório, e técnicas cujos erros sobre o gradiente descendente estocástico podem ser avaliadas como a quantização fixa e quantização usando planos de bits generalizados/representações ternárias.

Referências

[1] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys Tutorials*, 2020.

[2] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017*, volume 54, 2017.

[3] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

[4] Davide Proserpio, Sharon Goldberg, and Frank McSherry. Calibrating data to sensitivity in private data analysis: A platform for differentially-private analysis of weighted datasets. *Proceedings of the VLDB Endowment*, 7(8):637–648, 2014.

[5] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. *Journal of Privacy and Confidentiality*, 7(3):17–51, 2016.

[6] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proceedings of the 3rd MLSys Conference, MLSys 2020, Austin, TX, USA, March 2-4, 2020*, 2020.

[7] Tomoaki Nishio and Ryo Yonetani. Client selection for federated learning with heterogeneous resources in mobile edge. In *2019 IEEE International Conference on Communications, ICC 2019, Shanghai, China, May 20-24, 2019*, pages 1–7. IEEE, 2019.

[8] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[9] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: Communication-efficient SGD via gradient quantization and encoding. *Advances in Neural Information Processing Systems*, 30, 2017.

[10] Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Terngrad: Ternary gradients to reduce communication in distributed deep learning. *Advances in neural information processing systems*, 30, 2017.

[11] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signSGD: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pages 560–569. PMLR, 2018.

[12] Kai Liang, Huiru Zhong, Haoning Chen, and Youlong Wu. Wyner-ziv gradient compression for federated learning, 2021.

[13] Luke Melas-Kyriazi and Franklyn Wang. Intrinsic gradient compression for scalable and efficient federated learning. In Bill Yuchen Lin, Chaoyang He, Chulin Xie, Fatemehsadat Mireshghallah, Ninareh Mehrabi, Tian Li, Mahdi Soltanolkotabi, and Xiang Ren, editors, *Proceedings of the First Workshop on Federated Learning for Natural Language Processing (FL4NLP 2022)*, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[14] Venkata Gandikota, Daniel Kane, Raj Kumar Maity, and Arya Mazumdar. vqsgd: Vector quantized stochastic gradient descent. In *International Conference on Artificial Intelligence and Statistics*, pages 2197–2205. PMLR, 2021.

[15] Yuqing Du, Sheng Yang, and Kaibin Huang. High-dimensional stochastic gradient quantization for communication-efficient edge learning. *IEEE Transactions on Signal Processing*, 68:2128–2142, 2020.

[16] Xiufang Shi, Wei Zhang, Mincheng Wu, Guangyi Liu, Zhenyu Wen, Shibo He, Tejal Shah, and Rajiv Ranjan. Dataset distillation-based hybrid federated learning on non-iid data, 2024.

[17] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 25, pages 1097–1105. Curran Associates, Inc., 2012.

[19] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Lingyun Wang, and Jianbing Chen. Recent advances in convolutional neural networks. *Pattern Recognition*, 77:354–377, 2018.

[20] Rubing Xue, Jiaming Pei, and Lukun Wang. Federated learning on small batch sizes via batch renormalization. In *The Second Tiny Papers Track at ICLR 2024*, 2024.